**import** pandas **as** pd

**import** numpy **as** np

In [3]:

df **=** pd**.**read\_csv('sales\_data\_sample.csv', encoding**=**'unicode\_escape')

In [5]:

df**.**head()

df**.**info()

*#Columns to Remove*

to\_drop **=** ['ADDRESSLINE1', 'ADDRESSLINE2', 'STATE', 'POSTALCODE', 'PHONE']

df **=** df**.**drop(to\_drop, axis**=**1)

In [8]:

*#Check for null values*

df**.**isnull()**.**sum()

df**.**dtypes

*#ORDERDATE Should be in date time*

df['ORDERDATE'] **=** pd**.**to\_datetime(df['ORDERDATE'])

In [12]:

*#We need to create some features in order to create cluseters*

*#Recency: Number of days between customer's latest order and today's date*

*#Frequency : Number of purchases by the customers*

*#MonetaryValue : Revenue generated by the customers*

**import** datetime **as** dt

snapshot\_date **=** df['ORDERDATE']**.**max() **+** dt**.**timedelta(days **=** 1)

df\_RFM **=** df**.**groupby(['CUSTOMERNAME'])**.**agg({

'ORDERDATE' : **lambda** x : (snapshot\_date **-** x**.**max())**.**days,

'ORDERNUMBER' : 'count',

'SALES' : 'sum'

})

*#Rename the columns*

df\_RFM**.**rename(columns **=** {

'ORDERDATE' : 'Recency',

'ORDERNUMBER' : 'Frequency',

'SALES' : 'MonetaryValue'

}, inplace**=True**)

df\_RFM**.**head()

*# Divide into segments*

*# We create 4 quartile ranges*

df\_RFM['M'] **=** pd**.**qcut(df\_RFM['MonetaryValue'], q **=** 4, labels **=** range(1,5))

df\_RFM['R'] **=** pd**.**qcut(df\_RFM['Recency'], q **=** 4, labels **=** list(range(4,0,**-**1)))

df\_RFM['F'] **=** pd**.**qcut(df\_RFM['Frequency'], q **=** 4, labels **=** range(1,5))

df\_RFM**.**head()

*#Create another column for RFM score*

df\_RFM['RFM\_Score'] **=** df\_RFM[['R', 'M', 'F']]**.**sum(axis**=**1)

df\_RFM**.**head()

**def** rfm\_level(df):

**if** bool(df['RFM\_Score'] **>=** 10):

**return** 'High Value Customer'

**elif** bool(df['RFM\_Score'] **<** 10) **and** bool(df['RFM\_Score'] **>=** 6):

**return** 'Mid Value Customer'

**else**:

**return** 'Low Value Customer'

df\_RFM['RFM\_Level'] **=** df\_RFM**.**apply(rfm\_level, axis **=** 1)

df\_RFM**.**head()

*# Time to perform KMeans*

data **=** df\_RFM[['Recency', 'Frequency', 'MonetaryValue']]

data**.**head()

*# Our data is skewed we must remove it by performing log transformation*

data\_log **=** np**.**log(data)

data\_log**.**head()

*#Standardization*

**from** sklearn.preprocessing **import** StandardScaler

scaler **=** StandardScaler()

scaler**.**fit(data\_log)

data\_normalized **=** scaler**.**transform(data\_log)

data\_normalized **=** pd**.**DataFrame(data\_normalized, index **=** data\_log**.**index, columns**=**data\_log**.**columns)

data\_normalized**.**describe()**.**round(2)

*#Fit KMeans and use elbow method to choose the number of clusters*

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** sns

**from** sklearn.cluster **import** KMeans

sse **=** {}

**for** k **in** range(1, 21):

kmeans **=** KMeans(n\_clusters **=** k, random\_state **=** 1)

kmeans**.**fit(data\_normalized)

sse[k] **=** kmeans**.**inertia\_

plt**.**figure(figsize**=**(10,6))

plt**.**title('The Elbow Method')

plt**.**xlabel('K')

plt**.**ylabel('SSE')

plt**.**style**.**use('ggplot')

sns**.**pointplot(x**=**list(sse**.**keys()), y **=** list(sse**.**values()))

plt**.**text(4.5, 60, "Largest Angle", bbox **=** dict(facecolor **=** 'lightgreen', alpha **=** 0.5))

plt**.**show()

*# 5 number of clusters seems good*

kmeans **=** KMeans(n\_clusters**=**5, random\_state**=**1)

kmeans**.**fit(data\_normalized)

cluster\_labels **=** kmeans**.**labels\_

data\_rfm **=** data**.**assign(Cluster **=** cluster\_labels)

data\_rfm**.**head()